Dynamic Airport Configuration and Resource Scheduling

Jesse Clayton[*] and Brian Capozzi, Ph.D.[†]
Metron Aviation, Inc., Herndon, VA, 20170

   Increasing traffic demand has prompted the construction of additional runways and taxiways in the National Airspace System (NAS). As airport taxi networks become more complex, the difficulty of managing these operations often results in surface congestion and delays. New technologies are needed to help manage these airport surfaces. Typically, airport capacity is described in terms of the tradeoff in the use of the airport runways – a Pareto-optimal curve represents the set of optimal tradeoffs. In this paper, a MaxFlow algorithm is used as the basis for determining instantaneous capacity states for an airport described as a network. This generalizes the notion of capacity to include the "end-to-end" (runway to gate) airport throughput. Results obtained in applying MaxFlow to several airports are compared with Pareto curves. Issues of "reachability" of each capacity state (or configuration) and "workload" required to transition between arbitrary states are investigated so as to best match the network configuration (direction of flow) to a time-varying demand. A simple scheduling heuristic is introduced to implement airport-wide configuration states and the results of several experiments determine the feasibility of the approach.

Nomenclature

G	=	graph or network
E	=	set of edges contained within a graph
e_k	=	the k-th edge contained in a given path through a graph
V	=	set of vertices contained within a graph
$\pi(v_0, v_f)$	=	set of paths through a network from a designated initial vertex to a final vertex
π_m	=	a single path through a network
C	=	set of scalar capacities on a graph
F	=	tradeoff functions for bi-directional capacity in a graph
f	=	a specific tradeoff function for an edge or path in the graph
u_{x_cap}	=	the capacity in the forward direction on an edge or path
v_{x_cap}	=	the capacity in the backward direction on an edge or path
u_x	=	the allocation in the forward direction on an edge or path
v_x	=	the allocation in the backward direction on an edge or path
Δu	=	the interval between which to sample departure allocations
c	=	the capacity of a resource
T	=	a time period parameter
t	=	a time parameter, such as a resource occupancy time
v	=	a velocity parameter
d	=	a separation distance parameter
n_x	=	a parameter indicating a number of operations
l	=	a length parameter
$x_{\leftarrow}$	=	a subscript to distinguish between the directionality of operations
γ	=	a cost function

---

[*] Software Analyst, R&D Division, clayton@metronaviation.com
[†] Senior Analyst, R&D Division, AIAA Member, capozzi@metronaviation.com

I.   Introduction

TRAFFIC Flow Management (TFM) personnel have the challenging task of balancing capacity with demand for the resources of the National Airspace System (NAS). Generally speaking, under current operations the runways at congested airports represent the most significant constraints on the system[1,2]; thus, improvements to airport runway throughput are predicted to result in an overall NAS capacity increase. In its Operational Evolution Plan[3] (OEP), the Federal Aviation Administration (FAA) is addressing this issue in several ways, including the construction of additional runways, high-speed exit ramps, reduced separation standards, and special maneuvers such as Land And Hold Short Operations (LAHSO). As airport surfaces and operations become more complex, and runways more efficient, the runway capacity may no longer be the limiting factor on NAS operations, and capacity estimates based on the runway performance alone may be insufficient for NAS planning. Furthermore, dynamic allocation of airport resources may be required in order to realize intended capacity improvements on such complex topologies. Towards this end, this paper presents the MaxFlow algorithm and its potential use in helping operational personnel to estimate capacity and allocate demand for airport taxiways and runways.

Currently, in order to operate under different weather conditions or to meet environmental constraints (e.g. noise abatement), personnel at an airport select from a set of pre-defined runway configurations to manage the airport. These configurations dictate the runways used, in which directions, and for what operations (i.e. arrivals or departures, or a mix thereof). Each runway configuration is generally associated with a capacity value pair indicating the estimated arrival and departure capacity of the airport under such conditions. The method used by the FAA for developing airport capacity estimates is based on testimony of airport personnel, historical arrival and departure records, and a combination of analytical and computer airport capacity models[4,5]. These values, particularly the anticipated airport acceptance rate (AAR), are used by TFM for planning NAS-wide initiatives. The set of all such capacity pairs can be used to define a discrete theoretical tradeoff function between arrival and departure capacity achievable at a given airport. Drawing a line through the outer edge of all such points can then estimate a Pareto optimal tradeoff curve for the airport. While this curve may aid in estimating the demand that the airport is capable of handling, however, it does not provide operational personnel with any information describing *how* to operate the airport to realize the theoretical capacity. Further, there is no guidance regarding operating the airport at points other than the points mapped to the discrete set of pre-defined airport configurations. Many larger airports define one or more standard taxi "plans" that describes how the major taxiways are to be used in order to best support a given runway configuration[6]. Such taxi plans are often developed through the use of detailed airport operation simulators such as TAAM or SIMMOD. In practice, however, airport personnel have few tools at their disposal to assist in intelligently changing the allocation of taxiways and runways in order to improve airport efficiency under changing conditions. The Surface Management System (SMS)[7], a NASA-funded decision-support tool (DST) currently transitioning to the FAA for deployment, is a first step toward providing such information. SMS has the ability to provide controllers in the tower with suggestions regarding runway configurations (and/or the time at which a given configuration should occur) that can best serve the anticipated demand. Further, SMS can provide estimates to NAS users of when runway configuration changes are likely to occur to assist in planning their gate and ramp operations. However, a gap still exists in terms of the availability of information related to how best to route traffic on the airport surface network in order to realize efficient operations under the suggested configuration change.

In this paper, the MaxFlow algorithm is used to determine many (millions) of capacity states for an airport, and to generate the corresponding allocation of resources. The algorithm does not rely on empirical throughput data, and thus may be applied without extensive historical test cases. We also describe a means for evaluating the demand-independent cost of switching from one capacity state to another in order to facilitate selection of airport configurations in a dynamic environment, and evaluate the feasibility of a subset of the generated capacity states with an airport simulator.

II.   The MaxFlow Algorithm

The MaxFlow algorithm allows for the capacity to be represented as a Pareto-optimal curve between departures and arrivals, rather than a simple number of operations. Additionally, the algorithm generates an allocation of capacity to individual paths through the airport to achieve such maximum flow.

A. Airport Graph Representation

The airport surface is modeled as a graph G, where the vertices V in G represent taxiway/runway intersections and the edges E represent runways and taxiways. Each independent runway is represented by an edge, and each such runway edge is connected to a single vertex v_f representing a sink node in the network. The point at which taxiing

aircraft are typically handed off from the ramp area to the local controller, or the "spots", are represented as vertices, each of which is connected by an edge to a single vertex v_0 representing the a source node in the network. The designation of source vs. sink in this method is unimportant, as capacity is allocated for both arriving and departing aircraft. The graph for Newark International Airport (EWR) is provided as an example in Figure 1.
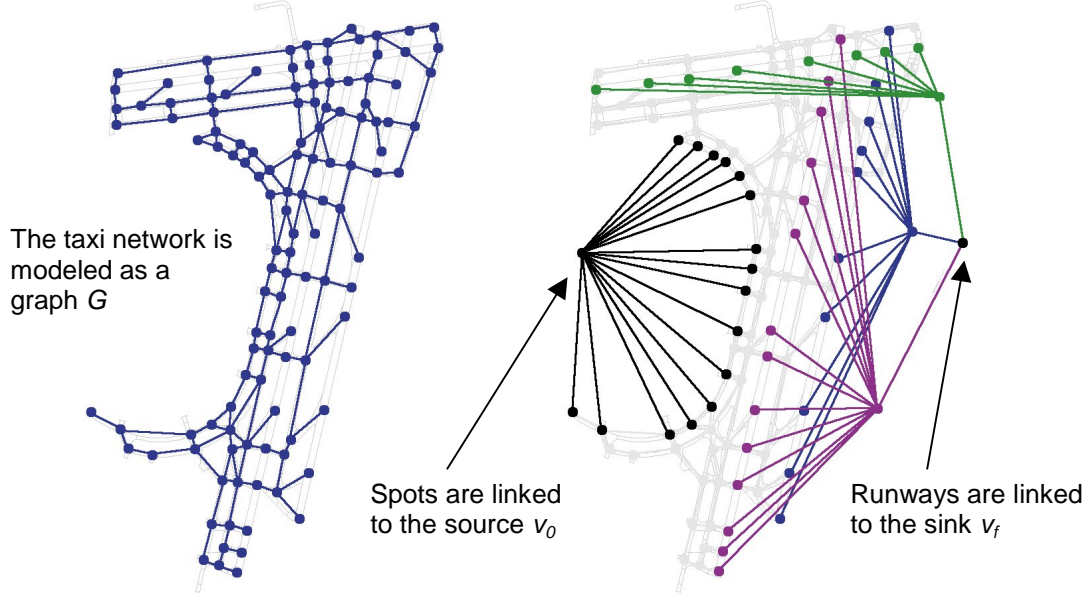

The taxi network is modeled as a graph G

Spots are linked to the source v_0

Runways are linked to the sink v_f

Figure 1: The graph representation for EWR

In a typical max-flow, a set of scalars C is used to represent the capacity of the edges in E, and an algorithm such as that presented by Ford and Fulkerson[8] is used to develop a flow allocation that provides the maximum flow sustainable by G. Inherent in airport surface operations, however, is the tradeoff between arrival and departure capacity. Surface resources (taxiways and runways) provide room enough for traffic to flow in one direction, and there is typically a significant cost associated with alternating flow direction, particularly in the case of longer segments. For this reason, the assignment of simple capacity values to each edge in the network is insufficient to describe surface operations.

Rather than a scalar capacity value, tradeoff functions F are developed that define the relationship between flow in one direction vs. flow in the opposite direction over a particular edge in E. F is non-increasing over its domain. Such a function is also associated with each unique acyclic path $\pi(v_0,v_f)$ in G. The derivation of F for individual resources is described later. Several iterations of a modified Ford-Fulkerson algorithm are conducted on G, iteratively selecting a departure flow allocation for each $\pi(v_0,v_f)$ and evaluating the remaining arrival capacity[‡]. According to Ford-Fulkerson, the flow allocated to an individual path is equal to the flow that the most constrained edge in the path can handle. Additionally, we consider the capacity associated with the path $\pi(v_0,v_f)$ as a whole. This is necessary because in a bi-flow operation, aircraft moving in one direction on a path must be held until the path is clear of aircraft moving in the other direction on the path. Individual nominal capacity pairs (allocation in a forward direction and in a backward direction) per edge and per path are given as (u_{x_cap},v_{x_cap}), where $v_{x\_cap} = f_x(u_{x\_cap})$. The allocated capacity on each edge and path is tracked through the execution of the algorithm as (u_x,v_x). Thus, each path is allocated capacity for both arrivals and departures. Each allocation iteration effectively evaluates the maximum bi-directional flow on the same graph for different mixes of arrivals and departures, and the purpose of the iterations is to evaluate many possible capacity states over G, given that there exists a tradeoff F for each edge and path. The MaxFlow algorithm is presented next.

[‡] The decision to set a departure allocation and then evaluate the arrival capacity rather than the other way around is arbitrary. Switching these operations has no effect on the algorithm performance or results.

MaxFlow Algorithm:

```
allocate(πₘ) :                          Definition of the recursive allocation function
    for i := 0,1...,n                    Iterate over departure allocations
        capₘ:= eₓ∈πₘ, min(fₘ (uₓ+ iΔuₘ)-vₓ)   The remaining capacity is limited by the path tradeoff
                                        and any existing edgewise allocations
        if capₘ ≥ 0                      If there is remaining capacity, allocate it
            for eₓ∈πₘ                    Allocate each edge in the path
                uₓ := uₓ+ iΔuₘ           in the forward direction,
                vₓ := vₓ_+ capₘ         and in the backward direction.
            end for
        end if
        if πₘ= πf                       If this is the last path in the list,
            output_current_capacity_state   output the capacity state
        else
            allocate(πₘ₊₁)              Begin allocation of the next path in the list
        end if
        unallocate(πₘ)                  Back out the allocation,
    end for                             and iterate on the next departure allocation
end

MaxFlow(G,F) :                          The MaxFlow algorithm allocates paths in G,
    allocate(π₀)     where π(v₀,vf)∈G   starting with the first one selected
end
```

The result of the algorithm is many distinct capacity states for an airport, which are defined by arrival and departure rates and a flow configuration for the airport surface. Because of the large number of possible states (x^n, where x is the number of points evaluated for each path and n is the number of paths evaluated), the algorithm produces an unmanageable amount of output, even for a relatively small graph. Only the capacity states resulting in the highest capacities are kept. Example results are provided later.

B. Defining Directional Tradeoffs

Crucial to this method of building airport surface capacity tradeoff curves is the assignment of directional tradeoff curves to each of the edges and the path in the defined network. Here the method by which these functions are approximated is presented. Surface resources are categorized by the way in which they handle demand. A runway, for example, likely has different constraints that define its capacity than does a taxiway. For each category of resource the first order parameters that constrain this capacity are presented, and the way in which they define individual tradeoff curves. Improved second order parameters that contribute to this effect are also suggested.

1. Independent Runways

Approaches and takeoffs are fundamentally limited by the Miles-in-Trail (MIT) separation imposed by wake vortex restrictions of aircraft.

Figure 2: Wake vortex restrictions d_{min} limit approach and departure capacities

The capacity due to such constraints is given by equations 1 and 2, where T is the time period of interest:

Approach/Departure Paths

$$C_{arr} = \frac{T \cdot v_{nom\_app}}{d_{min\_app}} \qquad (1)$$

$$c_{dep} = \frac{T \cdot v_{nom\_dep}}{d_{min\_dep}} \tag{2}$$

Different aircraft types (and ordering of aircraft types in a particular stream) have different MIT requirements for wake vortices. Furthermore, even identical aircraft operating in different wind conditions have differ in nominal groundspeeds for takeoff and landing. Such level of detail is not modeled; minimum separation distances and nominal speed averages are used that account for an assumed fleet mix.

In addition to minimum spacing, the dual use of the runway depends on the runway occupancy time for each operation. Arrival capacity is stated in terms of the number of departures, and departure capacity stated in terms of the number of arrivals.

Independent Runways

$$c_{arr} = \left(1 - n_{dep}\left(\frac{t_{dep\_occ}}{T}\right)\right) \cdot \left(\frac{T}{t_{arr\_occ}}\right) \tag{3}$$

$$c_{dep} = \left(1 - n_{arr}\left(\frac{t_{arr\_occ}}{T}\right)\right) \cdot \left(\frac{T}{t_{dep\_occ}}\right) \tag{4}$$

Since a runway tradeoff function F is subject to all of these constraints, the capacity of a runway is defined as the intersection of the areas under these functions. This is shown in Figure 3. Values lying outside the first quadrant are not of interest.
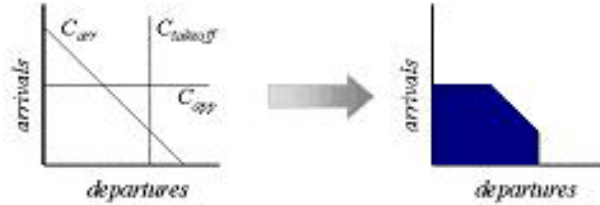

Figure 3: Building the tradeoff curve for an independent runway

2. Dependent Runways

Several types of dependencies should be considered when runways are co-constrained. These are presented in Figure 4.

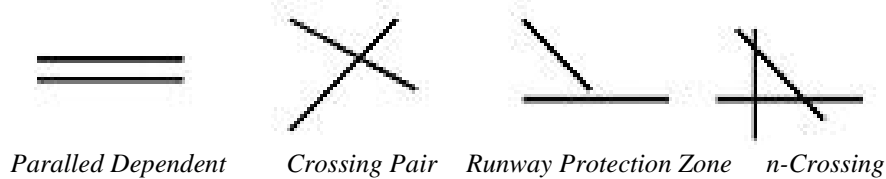*Paralled Dependent   Crossing Pair   Runway Protection Zone   n-Crossing*

Figure 4: Runway dependencies.

The effective capacity for each combination may be reduced from the sum of the independent capacities by the dependency. In many cases, these reductions turn out to be procedural rather than physical. For example, in current operations parallel dependent runways are often treated as a single runway if one is being used strictly for arrivals and the other for departures. In Visual Flight Rule (VFR) conditions, the runways may behave independently with the caveat that they are approved for closely spaced parallel approaches or simultaneous departures, and operations must not violate wake vortex restrictions. These cases are represented by the following equations.

Parallel Dependent Runways

$$c_{\parallel arr}(n_{\parallel dep}) = c_{arr1}(n_{dep1}), c_{arr2}(n_{dep2}) \tag{5}$$

$$c_{\parallel arr}(n_{\parallel dep}) = \max(c_{arr1}(n_{dep2}), c_{arr2}(n_{dep1})) \tag{6}$$

Equations (5) and (6) show how procedure can be represented in the individual capacity tradeoff curves of each resource. Equation (5) states that, from a capacity standpoint, the parallel runways are treated independently. Equation (6) states that splitting the operations results in a reduced capacity comparable to a single runway. Incidentally, the resource represented by equation (6) would be treated as a single edge in the network representation, connected to multiple taxiway entrances and exits.

Crossing runways can also be treated as independent runways with the added constraint that there is a limited capacity at the crossing point. This crossing point becomes another edge in the network representation, whose tradeoff curve is given by equation (7). The same representation can be used for a runway protection zone dependency.

Crossing pair, runway protection zone

$$c_{Xarr} = \left(1 - n_{Xdep}\left(\frac{tx_{dep\_occ}}{T}\right)\right) \cdot \left(\frac{T}{tx_{Xarr\_occ}}\right) \tag{7}$$

This representation does not allow for the assignment of different occupancy times for the departures and arrivals on the specific runways. Such a level of detail is not likely to be significant in this representation. A similar technique could be used to capture the capacity of n-crossing runways, though such application has not yet been investigated.

3. Taxiways

Taxiways comprise the bulk of the surface network. For these resources, the notion that there is a tradeoff between arrivals and departures breaks down. The tradeoff is now one of directionality, and results from the necessity of holding aircraft in one direction so that some aircraft may travel in the opposite direction. Consider a stream aircraft from left to right and another single aircraft moving from right to left as in Figure 5.
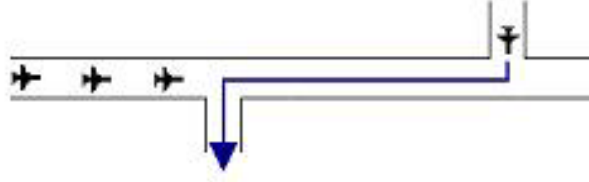

Figure 5: Bi-directional taxiway use

Aircraft moving in both directions are assumed to have similar taxi speeds and spacing requirements. The taxiway segment would achieve maximum throughput by allowing a minimally spaced stream of aircraft to move through in one direction. The cost associated with sending a single aircraft through in the opposite direction is holding the stream for the amount of time required to a) make room on the segment for the aircraft and b) for the aircraft to transit the segment. The tradeoff is given by equation (8), where v_{tx} refers to the nominal taxi speed, l_{seg} refers to the length of the individual taxi segment and d_{tx_min} refers to the minimum separation requirement. Again, different aircraft and even different airlines and pilots have different preferences for taxi speed and separation minima. Here an average is used. Note that the arrow subscripts are used to differentiate the direction of movement.

Taxiway movement

$$
\begin{aligned}
c_{tx \rightarrow}(n_{tx \leftarrow}) &= \frac{T \cdot v_{tx}}{d_{tx\_min}} & c_{tx \leftarrow} &= 0 \\
c_{tx \rightarrow}(n_{tx \leftarrow}) &= \frac{T \cdot v_{tx}}{d_{tx\_min}} - \left(\frac{2 \cdot l_{seg}}{d_{tx\_min}} + n_{tx \leftarrow}\right), & c_{tx \leftarrow}(0) &> c_{tx \leftarrow} > 0 \\
c_{tx \rightarrow}(n_{tx \leftarrow}) &= 0 & c_{tx \leftarrow} &= c_{tx \leftarrow}(0)
\end{aligned}
\tag{8}
$$

<u>Ramp Areas</u>

While the ramp area comprises a significant component of airport operations, the management of such is not currently treated in this technique. Though the ramp is typically operated as a set of taxiways, the expanse of concrete does not physically constrain the movement area as a taxiway or runway does. The complexity of possibly 2-dimensional resources (as opposed to a 1-dimensional taxiway or runway) may complicate this representation of the airport.

C. Algorithmic Enhancements and Simplifications

Because of its iterative nature, the MaxFlow algorithm explores many possible flow tradeoffs from source to sink. The processing time is approximately proportional to the number of times any element in $\pi(v0,vf)$ is evaluated. The total count of such evaluations is equal to $n^{count(\pi(v0,vf))}$. The number of possible paths through even a simple network is large, and for a network representing a typical airport surface, the time to solution is makes the problem computationally infeasible for the MaxFlow algorithm. To mitigate this problem, the number of paths through the network available to the algorithm is artificially limited at 25, and n is set to 3. The paths are selected so as to represent shorter distances while covering all runways and spots. First, the shortest path from an arbitrarily chosen spot-runway pair is selected. Next, the shortest path from the same spot to a different runway is selected. This continues until the runways are exhausted, at which point the shorted path from a new arbitrarily chosen spot-runway pair is selected. When the spots have been exhausted, the next shortest path from the first spot-runway pair is selected. The process continues until the count of paths is greater than 25.

The implication of the reduced number of paths is that some available capacity may remain unallocated when the algorithm terminates. The amount of unallocated capacity may be significant, particularly for larger and more complex graphs; this limitation prevented the analysis of Dallas-Ft. Worth International Airport (DFW) in the experiments presented later. The implication of the reduced number of iterations, n, is fewer resulting capacity states as well as a tendency of the remaining capacity states to deliver the same capacities.

III.   Algorithm Validation

In order to validate the MaxFlow algorithm, it was run on models of commercial airports and the resulting capacities compared to those reported by the FAA[9]. The parameters for the tradeoff functions F described in the previous section were selected to approximate separation standards under VFR conditions and average airspeeds and taxi speeds in historical Enhanced Traffic Management System (ETMS) data and airport surface surveillance data. In each case, certain runways were removed from the graph representation of the airport to reflect runway usage during the current operational VFR configuration, and edges connecting runways to the taxi network were restricted to operate in one direction to reflect the flow direction of the airport. Five airports were selected for this validation exercise. These were as follows:
- SAN – San Diego Intl-Lindbergh Field
- EWR – Newark International
- BOS – Gen Edward Lawrence Logan International
- MEM – Memphis International
- ORD – Chicago O'Hare International

The capacity states determined by the MaxFlow algorithm appear to agree with the FAA reported capacity benchmarks, in most cases within 20%. Results for two such experiments are presented in Figure 6. For most of the airports evaluated, data was not available for the taxiway configurations used by the operational personnel, and thus a comparison matching of a specific capacity state as determined by the MaxFlow algorithm with a capacity state used to achieve a point on the FAA benchmark was not possible. Note that each point on the graph generated by the MaxFlow algorithm actually represents several capacity states, all resulting in the same total capacity.

The researchers suspected that the runways were the limiting factor in each case of modeled operations in the validation exercise. This was confirmed by effectively setting the capacity of the edges representing runways to be infinite, and again evaluating the capacity of the modeled airports with the MaxFlow algorithm. In all five test cases, the capacity of the taxiway network was significantly greater, although the taxiway network capacity alone tended to be closer to the airport capacity for larger and more complex graphs. In the largest evaluated (MEM and ORD) the taxiway network capacity was greater by less than 20%. This suggests that adding additional runways without improving taxi operations may tend to saturate the taxiway network. Setting parameters and modifying the graph representations so as to model Instrument Flight Rule (IFR) operations (different runways were used and arrival and departure spacing was increased) was also tested with similar results.
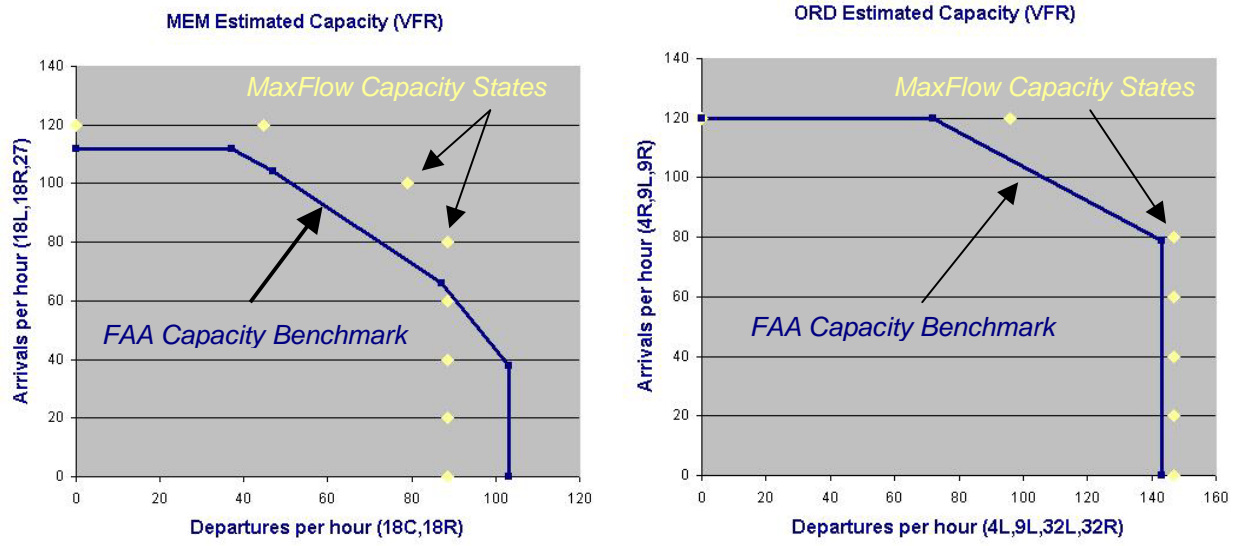
Figure 6: VFR capacity as estimated by the MaxFlow algorithm and the FAA Capacity Benchmark

There are several possible explanations for the relatively good agreement between the capacity as estimated by the MaxFlow algorithm and the FAA capacity benchmark. As suggested previously, the runways tend to drive airport capacity in the current system, and roughly modeling their behavior as a constraint is not difficult. While the feasibility of using the network in the manner proposed by the algorithm is investigated later in this paper, future research may be required to assess the capability of the MaxFlow algorithm to evaluate taxiway capacity of the existing system.

IV. Capacity States

A benefit of this approach to determining airport capacity is that the MaxFlow algorithm also provides information as to how the airport resources should be used in order to attain an estimated capacity. This information is a natural output of the execution of the algorithm. Since the implementation is abstract and can be used for any graph for which tradeoff functions are defined, the modification, addition, or removal of resources (runways or taxiways) is easily modeled. Such a capability allows for planning for high-capacity operations under abnormal conditions, such as resource closures, to provide airport flow plans for off-normal or novel runway configurations. Three airport surface configurations for generated MEM capacity states are presented in Figure 7.

One of the implications of allocating capacity in the method used by the MaxFlow algorithm is that the first few paths selected for allocation tend to exhaust the taxi network capacity and block allocation on paths originating at other spots. Operationally, personnel tend to use the spot located near the parking gate for individual aircraft, or one of the spots associated with the airline's ramp area. While such a mapping may be obvious when viewing the capacity states, there currently exists no way to generate it through the MaxFlow algorithm. Possible implementations are discussed in the future work section.

The issues described are evident in the graphical depiction of the MEM capacity states. In all three states, the main terminal is allocated only one spot from which to enter or exit the network, and in most cases the path does not represent a nominal route from the terminal to the runway.
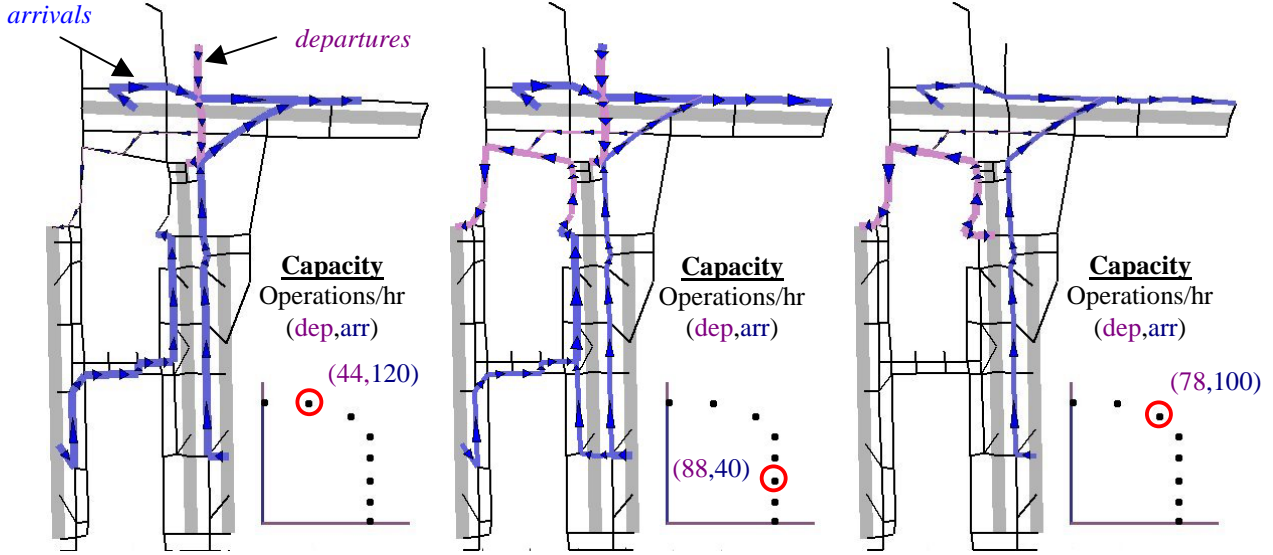
Figure 7: Several capacity states for MEM (arrival and departure capacity, along with the resource allocations is shown).

V. Capacity State Changing

In operations it is often necessary to change the configuration of the airport in order to handle a varying demand, operate under changing environmental conditions (e.g., shifting wind conditions), or meet procedures such as noise abatement. Factors that require unexpected changes temporarily reduce airport capacity, as aircraft have already been moved to arrive or depart on a runway that is no longer active. As presented earlier, the MaxFlow algorithm generates several capacity states that each lie along the Pareto-optimal frontier. In this section a simple cost function for changing capacity states is described.

The capacity reduction during a state change occurs primarily as a result of departing aircraft that are taxiing for a departure runway that has been deactivated, and arriving aircraft that are being vectored in the terminal airspace for an arrival runway that has been deactivated. Practically speaking, common factors that contribute to the real cost involved in such a situation include the cost of delays to airlines, the cost of fuel, and the additional workload of ATC personnel in moving and vectoring aircraft. In some extreme cases, arrival aircraft may not be able to absorb air delay and must divert to an alternate airport. These factors are in turn affected by the current state of surface operations, including the position of aircraft, congestion levels, and the difference in topologies and flow rates between the individual configurations or capacity states. Operationally, ATC personnel often try to change the airport configuration during a period of low demand, so as to mitigate the effect of congestion.

The cost function presented here assumes that demand is at capacity. Given a flow direction, every capacity state represents some allocation of the same set of paths. In order to free capacity for operations that exist in a new configuration, but not in the present configuration, aircraft must be delayed on paths. The delay occurs until the last aircraft not affected by the configuration change completes its traversal of the taxiway network, and thus frees the path for new operations. The actual cost, then, is proportional to the product of the capacity reduction and the time taken to transit the path (the length of the path, assuming constant and uniform taxi speeds). Because of the subjectively high expense of keeping aircraft aloft or possibly forcing an alternate arrival airport, the cost of reducing the capacity of an arrival stream may be higher. The cost function for changing from one capacity state to another is as follows:

$$\gamma = v_{taxi} \sum_{\pi(v_0, v_f)} A \cdot l_{\pi i}(c_{arr\_i} - c_{arr\_f}) + D \cdot l_{\pi i}(c_{dep\_i} - c_{dep\_f}) \qquad (9)$$

where v_{taxi} is the nominal speed, $l_{\pi i}$ is the length of the ith path, c_{arr_i}/c_{dep_i} and c_{arr_f}/c_{dep_f} refer to the arrival/departure capacity of the paths before and after the configuration change, respectively. A and D are coefficients for weighting arrival and departure costs.

The cost function from and to the generated capacity states and a configuration change matrix is generated, mapping the cost of moving from one to the next. The configuration change matrix is useful in two ways: 1) Given a projected demand, only some capacity states are feasible, and the matrix allows operational personnel to select from the lower cost solutions. 2) if a specific capacity state is desired, the cost graph of moving from any other state is generated and the solution can be reached by formulating the Markov Decision Problem (MDP). Given a demand projection for an entire day, the plan for changing capacity states can be so formulated, and a policy for the day developed. As an example of the results, the cost matrix several MEM capacity states (Figure 7) is presented in Table 1. The costs in this graph are relative and equal to the anticipated delay in minutes under capacitated conditions as described by Equation 9. The arrival and departure coefficients (A and D, respectively) are set to 1.

Table 1: The relative cost matrix for capacity state changes at MEM

State number (dep,arr)	Final States								
Initial States	1 (88,0)	104305 (88,20)	104306 (88,40)	104307 (88,60)	460627 (88,80)	460631 (88,80)	80035 (78,100)	87043 (44,120)	87047 (44,120)
1 (88,0)	0	1930	1930	1930	1912	1912	1287	1471	1471
104305 (88,20)	1956	0	0	0	16	16	511	1159	1159
104306 (88,40)	2344	387	0	0	16	403	511	1159	1546
104307 (88,60)	2731	774	387	0	16	790	511	1159	1933
460627 (88,80)	3487	1565	1177	790	0	774	1268	1924	2698
460631 (88,80)	3930	2007	2007	2007	1216	0	2485	3141	1924
80035 (78,100)	3414	2612	2224	1837	1819	2593	0	830	1604
87043 (44,120)	3585	3247	2860	2473	2463	3238	818	0	774
87047 (44,120)	4028	3689	3689	3689	3680	2463	2035	1216	0

The trend is that for moving to a dissimilar configuration (i.e., one that has very different arrival and departure capacities), the cost is higher, whereas for moving to a similar configuration, the cost is lower. In some cases, for example, moving from capacity state number 104305 to number 104307, the cost is zero. Investigation of the capacity states shows that the configurations are identical, except that the latter has additional paths open for use. The most costly change, moving from state 87047 to state 1, results from closing all the departure paths in order to make room for arrival allocation.

VI. Scheduling

While the MaxFlow algorithm generates estimated throughput values for different capacity states, scheduling of individual aircraft is not performed. An airport surface simulator was used to evaluate the feasibility and estimated throughput of the generated states. The simulator ingests airport definitions, departure and arrival demand data, generates and de-conflicts trajectories using shortest estimated time paths through the airport network. Airport resources not allocated in individual capacity states were removed from the airport definitions. The simulator was allowed to use any gate area, runway, and arrival or departure fix, so long that is was accessible from the remaining definitions. While actual arrival and departure demand typically has constraints on both the arrival and departure fixes, and the parking gate, such a level of detail is not necessary for this exercise.

Several demand models were generated. Departures and arrivals were created using a uniform random distribution of pushback times and on times (i.e., touchdown times), given arrival and departure rates (r_d, r_a). Various arrival departures rates were used with different relative magnitudes. For each evaluated capacity state, only demand models that were estimated to capacitate the network, both in arrival and departure operations, were used. Thus, the simulated airport throughput was determined. The results of this experiment are presented in Figure 8.
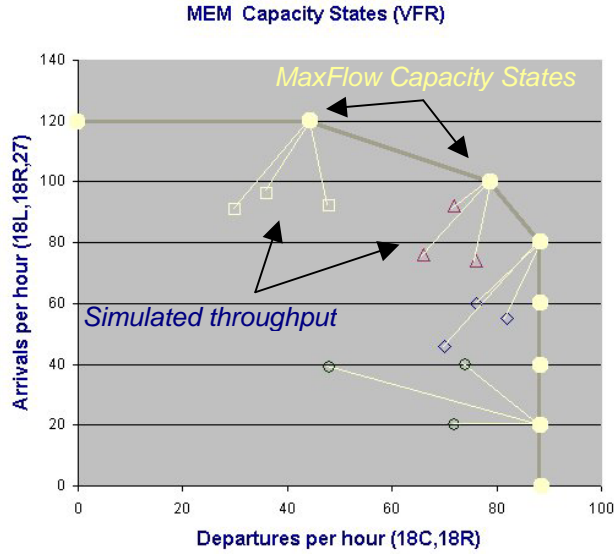
Figure 8: Generated MaxFlow capacity states and the simulated throughput of capacity states

The simulation results suggest that most of the tested capacity states are feasible, provided that they are scheduled in the right way. In general, the simulated throughput values are somewhat lower than the estimated capacities. This is primarily due to limitations of the simulator in that it generates the shortest time path for aircraft through the network, rather than the smallest delay path. While each aircraft may be scheduled such that it reaches a runway or parking gate soonest, this may have a detrimental effect on the airport operation as a whole, as aircraft will tend to cluster around inexpensive resources while other, though more costly resources, go unused. It is for this reason that demand increases in an already capacitated network may further increase throughput, as the shortest time paths begin to move away from the inexpensive, but saturated, resources. There were also issues of deadlock under some demand and capacity state combinations that were not resolved.

VII. Future Work

The MaxFlow algorithm and the analysis thereof can still be improved. As stated previously, the space in which the algorithm searches for allocations is artificially limited in order to generate a solution in a reasonable amount of time. Because shortest paths are favored for selection, and tend to occupy similar portions of the taxi network, the MaxFlow algorithm may overlook significant capacity regions. Allowing the algorithm to search a larger space might generate capacity states that result in greater capacity, or generate more states at the Pareto frontier, giving personnel additional choices when planning airport configurations. Since the solution time grows exponentially with the number of paths searched, increased computational power allows for only a marginal improvement in the feasible search space size. Instead, a method of selecting the set of paths that will generate the highest capacity states before the allocation step might be more useful. It is possible to run the algorithm several times on different subsets of the search space. Also, because of its iterative structure, the algorithm is easily separated for parallel processing.

The MaxFlow algorithm has the effect of allocating flow to the first path selected, thereby blocking other paths. This is problematic around the handoff points from the ramp area to the taxi network (i.e. the "spots") because it is not practical for every aircraft to enter the network from the same spot. Reducing the capacity of the resources representing spots or applying heuristics after execution to extend capacity allocated to a single path across many spots may mitigate this effect.

Development of the cost function to assess capacity state changes requires additional work. While the approach taken in this paper may provide insight into the relationship between capacity states under certain conditions, a cost function that explicitly accounts for the dynamic demand may be necessary. Research is also required to assess the accuracy of the cost function in evaluating the real cost to airlines and FAA personnel.

The capacity states explored here focused only on the use of certain runways in certain directions. A more complete representation of airport capacity may be generated by allowing different (and possibly arbitrary) combinations of runway usages, thus allowing personnel to match a capacity state to runway flow configurations

driven by environmental and procedural constraints. It is a relatively simple procedure to generate this data using the MaxFlow algorithm.

One implication of the generalized notion of capacity developed with the use of the MaxFlow algorithm is that capacity on specific airport resources may be equitably distributed according to the methods presented by Hall[10]. The method allows individual airlines to select their own operating point on a capacity region allocated to them, which in turn drives airport configuration selection. Such a system may provide a more collaborative approach to airport resource rationing.

Acknowledgments

References

[1]Hansman, R.J.: "The Dynamics of the Emerging Capacity Crisis in the US Air Traffic Control System," *MIT International Center for Air Transportation, Airline and National Strategies for Dealing with Airport and Airspace Congestion*, College Park, MD March 15-16, 2001

[2]Sawyer, B. et. al. "Surface Management System Airport Site Surveys" NASA CTO-05 CTOD 2 NASA Contract Number NAS2-00015, November 2001.

[3]Federal Aviation Administration (FAA) "Operational Evolution Plan" URL: http://www.faa.gov/programs/oep Feb 20, 2002

[4]Odoni, A. et al., "Existing and Required Modeling Capabilities for Evaluating ATM Systems and Concepts", Intl. Center for Air Transportation, MIT, March 1997.

[5]Federal Aviation Administration, "Airport Capacity And Delay", Advisory Circular 150/5060-5, 1983.

[6]Landrum & Brown, "Aircraft Operating Procedures Inventory: Chicago O'Hare International Airport and Los Angeles World Airport", December 11, 2002.

[7]Atkins, S., and Brinton, C., "Concept Description and Development Plan for the Surface Management System," *Journal of Air Traffic Control*, 2002.

[8]Ford, Fulkerson: *Flows in Networks* Princeton University Press, Princeton, NJ, 1962.

[9]Federal Aviation Administration (FAA), "Airport Capacity Benchmark Report, 2001", URL: http://www1.faa.gov/events/benchmarks/, FAA, 2001.

[10]Hall, W.: "Efficient Capacity Allocation in a Collaborative Air Transportation System," Ph.D. Dissertation, Massachusetts Institute of Technology, Cambridge, MA, 1999.